CS 240

Data Structures and Algorithms I

Alex Vondrak

ajvondrak@csupomona.edu

October 10, 2011

Homework 1 Submission

- Programming Projects
  - Not yet assigned
  - Submitted by email (no paper copy)
  - Substantial enough to warrant compiling/running
- Homework
  - Has been assigned
  - Submitted on paper
  - Can email proof you did the homework on a certain date
    - Must turn in identical paper copy next class session after your email
  - "But what about the programming-related questions?"
    - They're short; I don't want a digital copy
    - Write them by hand or print them out

A Correction

Previously said that, for constants k_1, k_2,

$$k_1 f(n) \underbrace{+k_2}_{\text{✗}} \in O(f)$$

Counterexample.

Fix k_1, k_2. Suppose $f(n) = \frac{1}{n}$.

$$k_1 \cdot 1/n + k_2 \in O(1/n)$$

$$k_1 \cdot 1/n + k_2 \leq c \cdot 1/n \qquad (\exists c > 0, n_0 > 0 \text{ and } \forall n \geq n_0)$$

$$k_1 + k_2 n \leq c$$

$$n \leq (c - k_1)/k_2$$

What's The Big Deal About Constants?

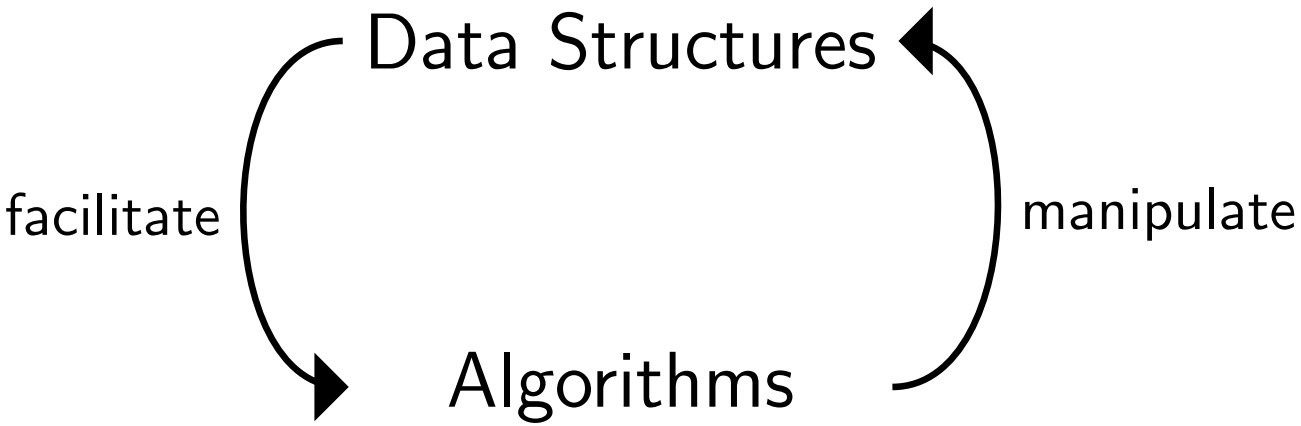
Intuitively, constants still "wash out" in the cases we care about—along with all the other less-significant terms

$$\begin{aligned} 10^{100} &\in O(1) \\ n^2 &\in O(n^2) \\ 2n^3 &\in O(n^3) \\ 1000n^5 - 400 &\in O(n^5) \\ 867 \cdot 2^n + n^2 - n &\in O(2^n) \end{aligned}$$

"Necessity is the mother of invention": O, Θ, Ω, o, ω, ...

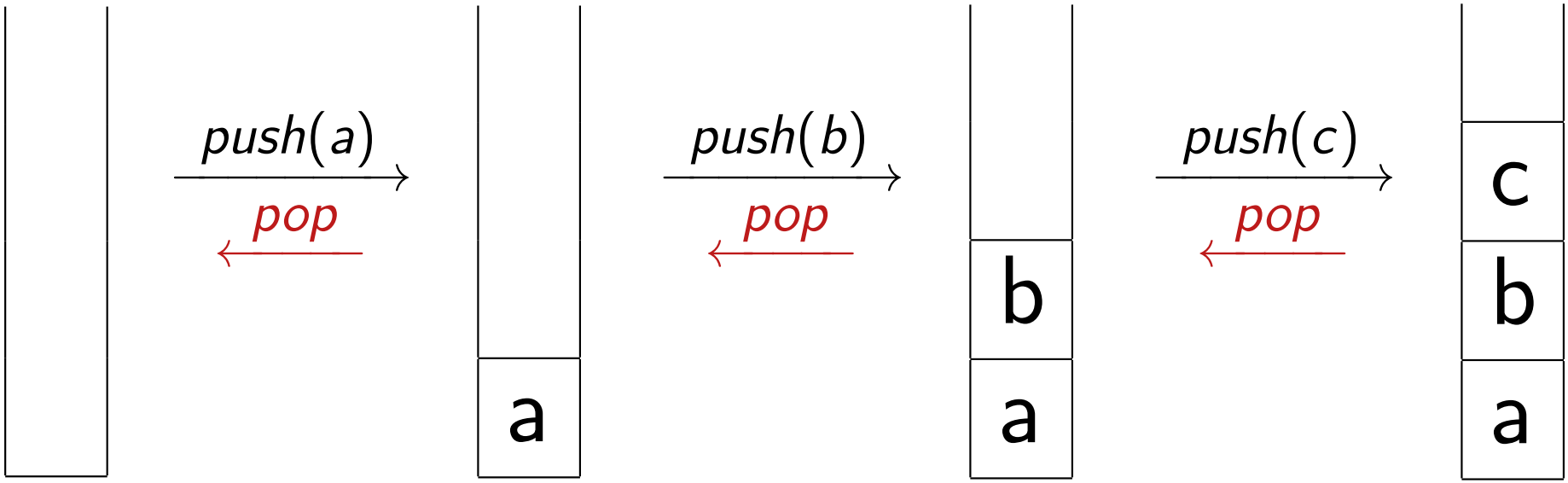
(cf. CS 331)

Data Structures



- In this class, we mostly study linear data structures
- Collections of items tend to have common operations
  - Adding elements
  - Removing elements
  - Querying for particular properties (membership, size, etc.)
- . . . But each operation raises its own questions

Stacks



Stacks

Methods

Stacks are defined by their insertion/deletion operators:

- `public void push(int item)`
- `public int pop()`

This makes stacks first-in, last-out (or FILO)

≡last-in, first-out ≡LIFO

Other common auxiliary methods:

- `public int top()` ⟵ or `peek()`
- `public boolean isEmpty()`
- `public int size()`

Stacks

Methods

Stacks are defined by their insertion/deletion operators:

- `public void push(int item)`
- `public int pop() throws StackUnderflowException`

This makes stacks $\underbrace{\text{first-in, last-out}}_{\equiv\text{last-in, first-out}}$ (or $\underbrace{\text{FILO}}_{\equiv\text{LIFO}}$)

Other common auxiliary methods:

- `public int top() throws StackUnderflowException`
- `public boolean isEmpty()`
- `public int size()`

Stacks

Abstract Data Type

```
interface Stack {
    public void push(int item);
    public int pop()
        throws StackUnderflowException;
    public int top()
        throws StackUnderflowException;
    public boolean isEmpty();
    public int size();
}

class SomeStackImplementation implements Stack {
    /* must implement all the methods */
}
```